Group Discussion Summary

Rick Schantz, (panel leader and briefer)

Gary Dougherty

Jim Hugunin

Mike Masters

Joe Loyall

Betty Cheng

Sally Howe

Bill Koenig

Steve Ray

Calton Pu

David Sharp

Joe Cross

Martin Rinard

Priya Narasimhan

Kane Kim

Cordell Green

Thuc Hoang

Eric Wohlstadter

Jason Scott

Premkumar Devanbu

Ira Baxter

Doug Schmidt

Outline

- New Forces/Opportunities/Requirements/Visions
- Example Applications
- Technical Problems
- Research Directions and Approaches
- Other Required Research
- Benefits or how we will make the world a better or safer place
- Observations on Approach
- What Could be Accomplished When
- Metrics of Progress

Forces and Visions

- Everything is a computer
- Everything is a networked computer
- Everything is potentially interdependent
- Things connect to the real world
- Increasing heterogeneity

Forces and Visions

- Complexity Threshold has arrived!
- **Fact:** Systems are growing and will keep growing
  - With growth comes increasing complexity and, thus,
  - A pressing need to keep application programming relatively independent of the complex issues of distribution and scale
- Inherent Complexities
  - discrete platforms
  - integration is the norm
  - partial failures are the norm
  - continuous operation and upgrade
  - changing environment and configuration
  - satisfaction of end to end properties in resource constrained environments
  - dynamic non-deterministic base

- Moving to Affordable and Dependable National Scope Critical Systems
- Consistent Experience over changing environments
- Save $ or construct larger systems as the dividend from a new approach to development
- A constant need to stay at the leading edge of knowhow
- A vaccine against software system failure
- Software catalytic converters to clean up the mess

Example Applications

- Integrated Medical Systems
- Terrorist Identification Systems
- Traffic control
  - Sensor data from 1000s of vehicles
- Swarms of UAVs
- National Voting System
- Theater battle management
  - varying granularities of coordination/missions in a hostile environment
- Supply chain management
- Community analysis of scientific data
  - Soft-real-time response and query optimization from 1000s of users, via coordinated management of 1000s of resources
- Home power management

Technical Challenges

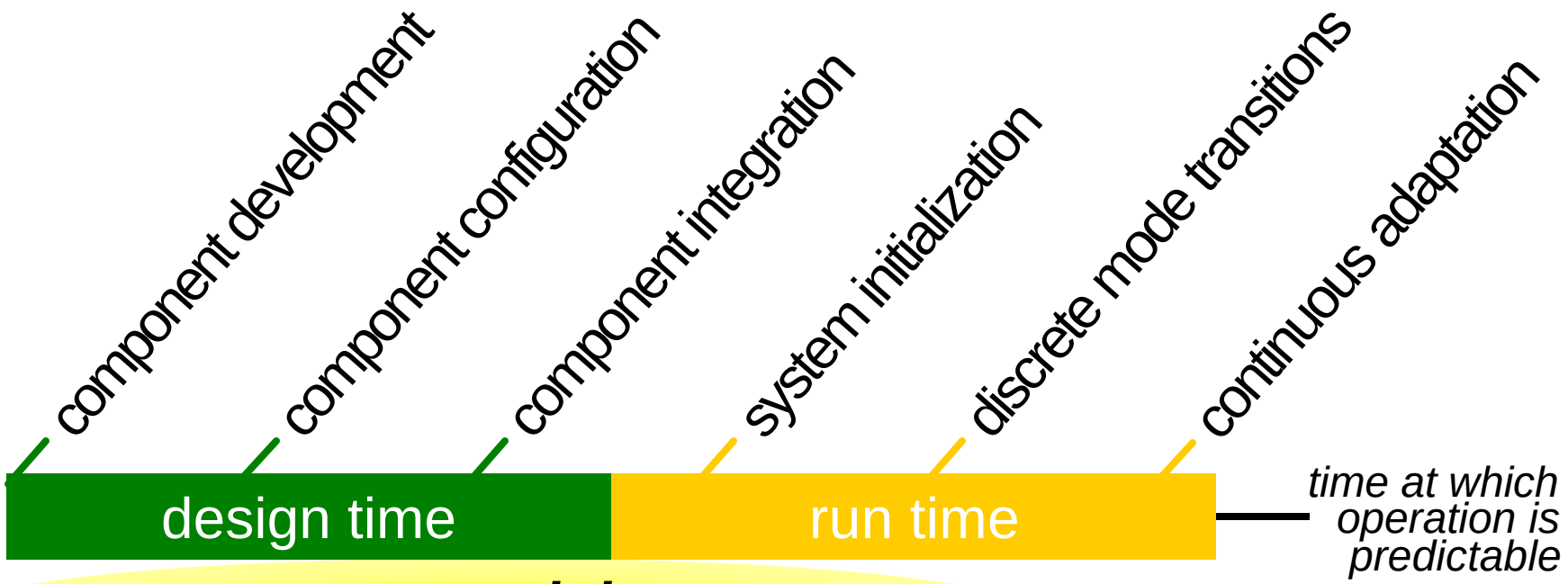
- Supporting resource management of multiple cross-cutting properties
  - Timeliness, quality, security, power, reliability, etc
  - Resource constrained (embedded) development and views: QoS, time/dependability/energy/footprint
- Lack of end-to-end properties in composite systems
- Lack of a computational model that allows for engineering tradeoffs
- Dynamic resource behaviors (in time and space): failure, variable load, changing requirements, ...
- Legacy: things that were not designed to work together now need to

Technical Challenges (continued)

- Risk, trust and control management
  - Policy/security/admin domains
  - Safety and validation of very dynamic systems
  - privacy
- Scale
  - Number of entities
  - Size of entities
  - Distance between entities
  - Number of entities composed in a single computation
  - Timescale over which network centric systems exist and non-stop behavior

Technical Challenges (continued)

Verifiability and Certifiability



current research leverages

current practice assumes

">$1B to certify"

Need To Extend Both Verification Timeline and Locality To Leverage Emerging Technologies

Compelling Research Directions and Approaches

- Four complementary thrusts that need to be addressed at all levels; one crosscutting/ coupling issue
- (1) View based projections that when combined deal with “the aggregate end-to-end problems”
  - New and flexible engineering tradeoffs
  - Operation at massive scales
  - Dynamic resource behaviors
  - Risk, trust and safety management

Directions and Approaches (continued)

- (2) Work on basic mechanisms that underlie “the problems” and the “views”
  - Resource tradeoffs: QoS mechanisms, RT, etc
  - Adaptive behavior
  - Scaling in various dimensions
  - Distributed control and coordination
  - adaptively using reflective (“own system”) information
- (3) Coordinated Multi-level resource management techniques
- (4) Construction of large systems with global behavior by composition of (small scale) network centric components; interoperability

View-Based Development

- Define Views
  - structural and behavioral (e.g. security, safety, rt perf., reliability, control, …,)
- Define Desired Analysis and Composition
  - Automating system configuration and generation of models
  - Predicting system level behaviors from local models
  - multi-dimensional tradeoff
  - global policy informs local behavior
  - Integration of views
- Develop Software Engineering Tools To Support All Above
  - define view representations
  - Define Automated Generation From Models/Analysis
  - Verify Conformance of Implementation to View

Other Required Research

- Development of metrics (benchmarks) to allow system developers to quantify (evaluate) the "-ilities" that their systems exhibit in practice
- Runtime Adaptability
- longer term evolvability
- (Automated) configuration and management of large-scale distributed applications

A Cut at Integrating Ideas

Future distributed systems will increase in size and complexity in order to meet the appetite for increased scope, competitive advantage and opportunistic interoperability deriving from easy connectivity

But number of interactions increases superlinearly with size, and overlapping attributes linearize, inadvertently customize and complexify the development process and product, making change impossible.

And number of interactions is proportional to schedule, cost, & defects[i.e. productivity], and to dependability and performance bounds [I.e. is it useful/useable]

Problem further exacerbated by distributed system characteristics, heterogeneous nature, intruders, nondeterministic substrate, …

Elements of a Solution:

1. **Factor problem specification into multiple, higher level, semantically sound, views to isloate complexity, reduce interactions, overlap, & inconsistencies, and promote change.**
2. **Populate these views with a quantifiable spectrum of varying cost solutions**

- **Provide automated support for sound *composition* of multiple views into composite high level specifications(models) while also facilitating tradeoff decision-making during the composition.**
- **Then provide automated support for the design and implementation of the specifications, via automated analysis of implementations and/or automated generation of implementations.**
- **Co-evolve the interconnected high level models and the implementations using the automated support toolset.**

Many research issues: is this feasible?, representation of specifications, composition methods, quantification of attributes, …

Benefits

- Building These Large Highly Distributed Systems Will Be More Affordable and Predictable and Safer…
- Societal
  - Build things that we just can't build now
  - Increase quality of future networked systems etc.: design by engineering not debugging
  - reduce exploding software development costs for the complex requirements of network centric systems in the real world
- Training
  - Seed industry (and academia) with people able to write new software & engineer new systems
- Better use of human resources
  - higher productivity
  - better match to people skills

Observations on Approach

- Teamwork is fundamental
  - cross-panel integration and result integration
  - large collaborative R&D effort is required to enable this large vision, to complement important individual efforts
  - industry buy in
- Need for large-scale projects to help us discover the real problems and validate partial results
  - common infrastructure
  - common challenge applications
- International collaboration is desirable

Schedule and Expectations

Metrics and Evaluation of Progress

- Transfer to real users
- commercial co-funding
- Discrete Experiments and Evaluation of Partial Results
- use challenge application to derive measures of success for the decomposition/composition technology

Collection of Ideas Raised During Group Discussion

In No Particular Order

- Aspects
- Better tools
- integration of various views --> running code
- informal techniques --> formal semantics
- adaptive middleware
- decomposition methodology & tools toward better blueprints
- tying implementation to the end effect result
- integrated properties & tradeoffs
- higher level RT abstraction
- distributed control
- invariant centric development practices
- global constraints transformed to local behavior, and dynamically recover from damage
- self-regulating software
- late binding